An Approach for Agent-Based Modeling Using AiC+

MARTIN NGOBYE and BENJAMIN KANAGWA[*]
School of Computing and Informatics Technology, Makerere University
THEO VAN DER WEIDE
Institute of Computing and Information Sciences, Radboud University Nijmegen
WOUTER T. DE GROOT
Institute for Science, Innovation & Society, Radboud University Nijmegen
CML, Leiden University, The Netherlands

ABSTRACT
In this paper, we present AiC+, an extension of the AiC framework designed for the explanation of human actions especially in the environmental field. We use first order logics to describe the semantics used to explain the action selection of the agent (actor) using an agent hierarchy system and a fuzzy typing relation. An example is illustrated using the AiC+ to validate the framework and discuss possible future extensions to the framework.

Keywords: Action-in-context, Agent hierarchy, First order logics, Fuzzy typing relation, Involvement function.

IJCIR Reference Format:
Ngobye, Martin; Kanagwa, Benjamin; Van Der Weide, Theo and De Groot, T. Wouter. An Approach for Agent-Based Modeling Using AiC+. International Journal of Computing and ICT Research, Vol. 5, Special Issue , pp 80-89. http://ijcir.org/specialissue2011/article9.pdf

1 INTRODUCTION

Recent years have witnessed a wide spread-interest in the multi-agent system approach to modeling. Multi-agent modeling offers a variety of ways in which many existing complex behaviors can be modeled. The potential benefits of multi-agent modeling will only be fully realized, however, on a basis of a systematic approach towards analyzing, designing and implementing the agent models. While there are many useful models of agents and multi-agent systems, they are typically defined in an informal way and applied in an ad-hoc fashion.

This paper introduces an extension of the Action-in-Context (AiC) framework designed for the explanation of human actions, commonly in the environmental field [De Groot 1992]. Based on the concept of progressive contextualization [Vayda 1983], the idea of AiC is to start out from the action to be explained, then identify the (individual or collective) actors directly causing this action, then identify the range of options available to these

[*] Martin Ngobye and Benjamin Kanagwa. School of Computing and Informatics Technology, Makerere University.
Emails: mngobye@cit.mak.ac.ug and bkanagwa@cit.mak.ac.ug
-Theo Van Der Weide. Institute of Computing and Information Sciences, Radboud University Nijmegen. tvdw@cs.ru.nl
-Woutert Dr Groot. Institute for Science, Innovation & Society, Radboud University Nijmegen and CML, Leiden University, The Netherlands. w.degroot@science.ru.nl

International Journal of Computing and ICT Research, ISSN 1818-1139 (Print), ISSN 1996-1065 (Online), Vol.5, Special Issue, pp. 80-89 December, 2011.

"primary" actors and the motivations attached to these options, and then identify other ("secondary") actors and factors influencing these options and motivations, thereby putting the action in its relevant causal context without a priority bias towards any scientific discipline [Vayda 1999]. With that, AiC is a fully actor-based framework, which is a logical choice for explanatory work because actors, not systems, are the social entities that cause change directly.

The AiC framework has four interconnected components [Overmars et al. 2007]. The first is an often repeated "core element," comprising of the action, the actor, his options and his motivations. In [Elster 1989], the latter two are called "opportunities" and "desires" but the structure is of the same simplicity: in order to act, people must have both the capacity and the will to do so. The other components of AiC are elaborations of the core element. The second component, the "actors field" is an aspect of AiC that describes the chains of social influence (causality, power) that run from the primary actors outward to other actors. Moving from primary to secondary actors and further is the actor-based way of moving from proximate factors to underlying drivers. The next component is mixed freely with the preceding one and consists of a "deeper analysis" of the options and motivations of selected actors. The final component is called the "actor model," which defines how the actor evaluates the options and motivations to come to his decision.

This model has so far been used in the Mameluke framework [Huigen 2004] which offers the modeler a generic format to implement the interacting behavior of the modeled entities. This generic format is a hybrid of the traditional belief-desire-intention (BDI) architecture [Rao and Georgeff 1995], the agent group role (AGR) architecture [Ferber and Gutknecht 1998; Rouchier et al. 1998; Kendall 1999], and behavioral decision and action models [De Groot 1992]. In the Mameluke framework, the behavioral model of a cognitive agent, i.e., the agents rules, is structured in potential option paths (POPs) and potential option nodes (PONs). Formally, a PON is a transaction interface between an initializing agent and a recipient agent. A POP defines a sequence of PONs. As a group, the POPs represent a theoretical construct of agent behavior and decisions that the framework user wishes to explore.

The current version of the AiC model is still informal and therefore the objective of this paper is to propose a formal extension and additional features to the AiC which we have denoted as AiC+. The use of formalisms is appropriate since they allow unambiguous descriptions of complex systems and also provide mechanisms which enable the construction of reliable and robust models.

The AiC+ framework uses an agent hierarchy based on an agent typing system which provides a more flexible description mechanism. In the hierarchy, each instance has associated a most specific type that inherits all properties of its ancestor type with the option to redefine actions introduced in the ancestor type. There also exists a fuzzy typing system where the fuzzy relation describes the type of relation between an instance and the agent types defined in the agent hierarchy. In this sense, the AiC+ is a more expressive and applicable model that can be deployed to interdisciplinary domains.

Most current models are partial in expressiveness and usability. We emphasize that AiC^{+} is an appropriate model which entails both properties depending on the application domain that we describe. Many models either have the expressiveness but lack the usability in the practical sense, or are useful in practice but are not expressive, in that no properties are given to determine the behavior of the system as a whole.

The remainder of the paper is organized as follows. The structure of the Agent is described in section 2. In section 3 we present the semantics of agents in the model. The description language of the model is introduced in section 4. In section 5, we offer an example that is used to validate the model. Finally in Section 6, we conclude with a discussion of the analysis of the model, and state the future work for our research.

2 AGENT STRUCTURE

In this paper, we focus on the characteristics of the AiC+ agents in isolation. An AiC+ model introduces a set of agent types that are grouped in a type hierarchy. Agents are assumed to have a sensory system to evaluate the environment that is shared by all agents [DeLoach and Valenzuela 2007]. An instantiation of an AiC+ model is a set of agent instances (or agents for short), where each agent has its own state at each moment. We assume that communication between agents can only be effectuated by communication actions. We assume an environment (arena) for all agent activity.

A communication action passes a message to another agent typically referred to as direct communication. This communication action has five parameters described as the expression send (Receiver, Performative, Language, Ontology, Content) where Receiver is the name of the receiving agent, Performative is a speech act name (e.g., inform, request, etc.), Language is the name of the language used to express the content of the message, Ontology is the name of the ontology used to give a meaning to the symbols in the content expression, and Content is an expression representing the content of the message. The other kind of communication is the indirect one where the effect of the action on an environment is described as an external action [Dastani 2008]. An external action is supposed to change the state of an external environment. The effects of external actions are assumed to be determined by the environment and might not be known to the agents

beforehand. An agent thus decides to perform an external action and the external environment determines the effect of the action. The agent can know the effects of an external action by performing a sense action (also defined as an external action), by means of events generated by the environment. It is up to the designer to determine how the effects of actions should be perceived by the agent as described in [Dastani 2008].

2.1 The Agent Hierarchy

AiC$^+$ is based on a typed agent system, where we distinguish between an agent (actor or instance) and an agent type. The agent type describes the properties that are shared by agent instances of that type. Type hierarchies are introduced to provide for a more flexible description mechanism. In a type hierarchy, each instance has associated a most specified type, that inherits all properties of its ancestor type, with the option to redefine actions that have been introduced in an ancestor type. So basically, an agent instance is assigned a most specific type, and is also related to its ancestor type.

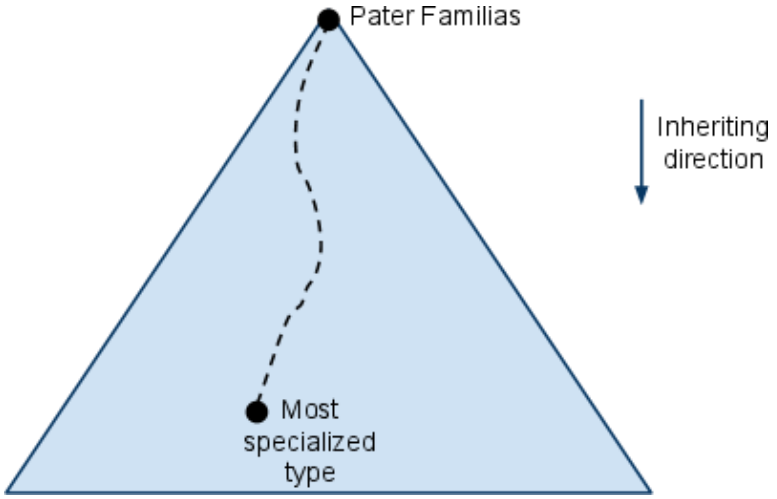

Fig. 1 Agent type hierarchy

In AiC$^+$ we have a more fuzzy typing system: there is a fuzzy relation describing the type relation between an instance and the agent types defined in some agent type hierarchy. So consequently, based on this fuzzy relation, an agent instance may make a fuzzy choice between actions that are available to all agent types in that hierarchy.

Let A be the set of agent types, then the agent hierarchy is described as the partial order (A, IsSpecOf). An agent type A is called a "pater familias" if it is not the specialization of another agent type: $\neg \exists_B [A \; IsSpecOf \; B]$. In the AiC$^+$ model, each agent A type must have associated its (unique) pater familias Π (A).

In the AiC$^+$ model, each agent A type must have associated its (unique) pater familias Π (A). We will call agent types A and B type related (A ~ B) when they have the same pater familias:

$$A \sim B \equiv \Pi(A) = \Pi(B)$$

We write HasType (a, A) to denote that agent a is an instance of agent type A. We will also refer to A as the base agent type of a. The fuzzy relation between an agent instance a and its related types then is expressed by the involvement function Inv where Inv (a, A) is the degree in which agent instance a is related to agent type A. We make the following assumptions:

1. each agent is involved in its base agent type:
   HasType (a, A) ⇒ Inv (a, A) > 0

2. each agent type is most involved in itself:
   HasType (a, A) ∧ B ≠ A ⇒ Inv (a, B) < Inv (a, A)

3. an agent is related to at most one agent type hierarchy:

Inv (a, A) > 0 ∧¬A ~ B ⇒ Inv (a, B) = 0

4. each agent is involved as least as much in its generalizations:
   Inv (a, A) ∧ A IsSpecOf B ⇒ Inv (a, B)

2.2 Agent Type Definition

An agent type is defined as a structure (*C, N, s_0, G,* Act). We subsequently describe these components.

C is set of the general conditions (boolean variables or attributes) that apply to an agent of that type. N is the set of variables (attributes) for that type of agent. From the two variables above, we build the set E of expressions, in the conventional way:

- from the numeric variables, we build relational numeric expressions
- from the boolean variables and relational numeric expressions, we build boolean propositions.

Some of the variables or values come from the inspection of the outside world (the macro environment), where the actor does not make a difference, while others are internal parameter settings of the agent (micro-environment) where the actor may have an impact on the physical or social environment. In each agent state, each variable has some value. Besides, each state may involve value assignments to variables from the agent type in which the agent is involved. States of agents of this type are determined by a value assignment to these variables. The agent type has initial state s_0.

The agent type also has an overall activity expression, G. This expression G specifies under what conditions the agent will be active at all. This may not be necessary as an explicit condition, as it may be integrated with the start condition for each action the agent may perform. Adding the overall activity condition G is more expressive which results in understanding the intention of the agent.

The component Act is a set of actions defined for that type of agent. The set Act is the set of actions specific for that agent type. We will use the expression TypeFrom (t) = A to denote that the action t is specific for agent type A. An action describes the behavior of an agent during a transition from one state to another. The AiC^+ agent has its goal as its benefit of action which is an overall benefit related to the motivation in the AiC^+ model. Each action is a tuple $\langle$ sc, bt $\rangle$ where sc is the start condition and bt is the benefit of that action expressed as a relational numeric expression. The start condition is a boolean proposition, which is an essential parameter used to evaluate whether the action can be triggered or not. The benefit parameter is the profit, so to say the goal the agent wants to achieve though it may not explicitly be stated in the model.

3 SEMANTICS

In this section, we present the semantics of the agent in a particular state. At any one moment, the agent is in a particular state which is defined as a value assignment to its variables. The agent then will select an action to perform. Preferably the agent will select an action from its associated agent type or ancestor type. But in the AiC^+ model the agent also may choose an action from any other agent type in its agent type hierarchy, depending on the level in which that agent is involved (at that moment) in that agent type. Therefore we introduce the extended action set $Act^+(A)$ for each agent type A as follows:

$$Act^+(A) = \bigcup_{x \in A} X.Act \qquad (1)$$

The applicability of each action in the agent extended domain is obtained from the involvement function Inv. The agent considers all actions in its extended set of actions. The selection of the next action for an agent of type A being in state s is done as follows. Find the actions t ∈ Act+(A) that are enabled,
The activity expression for the agent type associated with t is satisfied: s |= TypeFrom(t).G

The start condition of t is satisfied: s |= t.sc

This leads to a shortlist of actions $\Upsilon(a, s)$

$$\Upsilon(a, s) = \{ t \in Act^+(A) \mid s \models TypeFrom(t).G \wedge t.sc \} \qquad (2)$$

The shortlist is ordered according to the level of involvement and the benefit:

$t \preccurlyeq t' \equiv t.bt(s) \cdot Inv(a, TypeFrom(t)) \leq t'.bt(s) \cdot Inv(a, TypeFrom(t'))$ (3)

The operator $\preccurlyeq$ is referred to as the order of preference. So when an action has a high benefit, it will be considered even if the agent hardly is involved in the associated agent type. It may result in a change of involvement, when the agent decides to take another profession by changing its base agent type. In this paper we will not discuss changing of base agent type. Furthermore we assume that each agent has a special action called sense that only re-evaluates the environment leading to a modification of its state when a change of environment is observed.

3.1 Behaviour of the AiC+ Agent

Each agent instance has its unique possible execution, also known as the trace of action [Chainbi et al 1998]. The potential behavior of the agent is described by the set of all possible executions which are finite sequences of the form $s_0 \xrightarrow{t_1} s_1 \xrightarrow{t_2} s_2 \cdot$ This is inductively defined as:

1. If HasType (a, A), then $A.s_0$ is a possible execution of that agent with final state $st(X) = A.s_0$. X being a possible execution of an agent instance.
2. If X is a possible execution of agent a with HasType(a, A), and

   – st(X) |= A.G
   – t ∈ Y (a, st(X))
   – $st(X) \xrightarrow{t} s$

   then $st(X) \xrightarrow{t} s$ also is a possible execution of agent instance a with final state $st(X \xrightarrow{t} s) = s$

We will assume that the possible executions X and $X \xrightarrow{sense} s(X)$ are equivalent, and call X a reduced version $X \xrightarrow{sense} s(X)$. s(X) denotes the final state of the agent instance after a sense action on the environment. The set $ST(a)$ of all possible executions of agent instance a consists of all most reduced possible executions of that agent.

4. THE DESCRIPTION LANGUAGE

4.1 The Description Framework

Figure 2 describes the structure and features of the AiC$^+$ framework for describing an agent type. In box 1, we find the action which is eventually performed by the Actor. The Actor, labeled (box 2), takes a central position as the active element. The next level describes implementable options or simply put, what the actor can actually do, and the motivations (box 3.2) as the criteria through which the actor determines what implementable option he likes best. Similarly, the following level describes potential options defined as everything the actor knows how to do (box 4.1). In addition to these potential options, there is capital defined in (box 4.2) as the sum of all the resources the actor can access. Capital determines which of the potential options are implementable. Put together therefore, potential options and capital form the implementable options (box 3.1). Some of the motivational criteria of the actor are readily quantifiable, e.g in terms of money, hours, calories etc; these define the objectified motivations (box 4.3).

Other criteria act as multipliers (with values from 0 to higher than 1) on the objectified motivations; they are the degree to which the actor actually appreciates the objectified motivations; they are termed as interpretations (box 4.4). All arrows in the figure denote causal relations. The last level describes the interaction of the actor with the environment and other agents. It is at this layer that

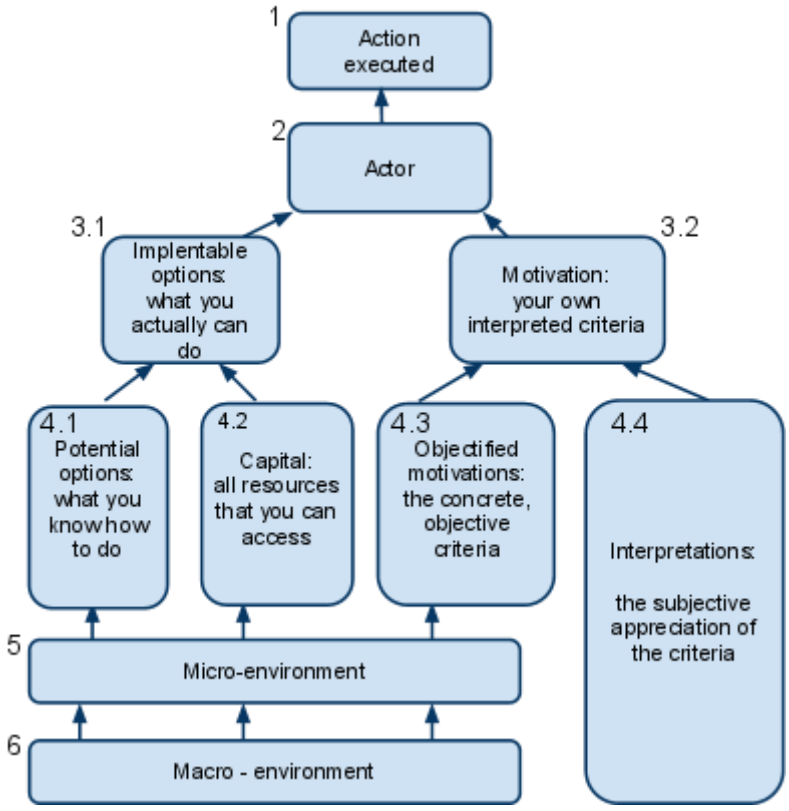

Fig. 2. The description framework

social and cultural features are described for instance the actions implemented are determined by ones status, or societal norms. Micro-environment (box 5) is any structure, physical or social, where the actor can has an impact for instance on his farm or his internet community. Macro-environment (box 6) is where the actor cannot make a difference, for example the oil market. For more information see [De Groot 1992]

4.2 The Specification Language

Using the specification language, one can formally specify what should be expressed as a structure of the agent and what should be written in each component of the AiC^{+} framework. We use the BNF notation to present the specification language. The specification language describes for each agent type in the following format:

Agent ⟨ ***Name*** ⟩
Specializes ⟨ ***Name SupertypeList*** ⟩ *
Attributes ⟨ ***AttributeList*** ⟩
Requires ⟨ ***Condition*** ⟩
Actions ⟨ ***ActionSpecList*** ⟩

We use ⟨ Name ⟩ to denote the name of the agent type. ⟨ Name SupertypeList ⟩ is the name of the ancestor or the general agent type. The ⟨ AttributeList ⟩ is the list of numerical variables and their dimensions as well as conditional variables. ⟨ Condition ⟩ is an expression used to specify under what conditions the agent type will be enabled, and ⟨ ActionSpecList ⟩ is the list of actions the agent can actually do.

4.3 The Semantics of the Specification

Describing the specification language using the AiC$^+$ framework enables us to derive how the agent determines its optimal choice out of actions in a shortlist, given a pool of options Act$^+$(A) . From the fuzzy typing system, the agent makes a choice between the actions that are available to all agent types in the hierarchy. Each agent type must have associated its unique pater familias, Consequently, the fuzzy relation between an agent instance and its related types is then expressed using the involvement function Inv. This is done because the agent considers all actions in its extended set of actions, eqn (1).

Using the motivations represented in (boxes 4.3, 4.4, and 3.2), aids the actor, given the capital in (box 4.2), to determine the implementable actions in (box 3.1). This is done only if the agent is enabled and that the actions are from the same agent type hierarchy. The implementable actions also referred to as the shortlist, eqn (2), are the actions the agent executes in that particular state. The actor then selects the best action from the shortlist using the level of involvement as well as the benefit, eqn (3), as seen in box 1. It can be the case that due to interaction with the environment by the agents using the sense action described in subsection 3.1, influences the choice the agent would take thereby having the need to re-evaluate the execution sequence of the agent.

5. THE EXAMPLE

We have described the behavior of how an AiC$^+$ agent evaluates its options and motivations in a given state and how the agent makes a fuzzy choice between actions that are available in an agent hierarchy. In this section we provide an example from the environment domain. The objective of the example is to illustrate how domain experts can use the model to explain the way actors in this environment would rationally make choices. This representation is done at a higher level of abstraction however most of the details are not included.

Central to the example are the agent types “farmer” and “fisherman” who specialize the general agent type or ancestor “person”. All persons can farm or fish. In our example, we only show the farmer description and leave out the

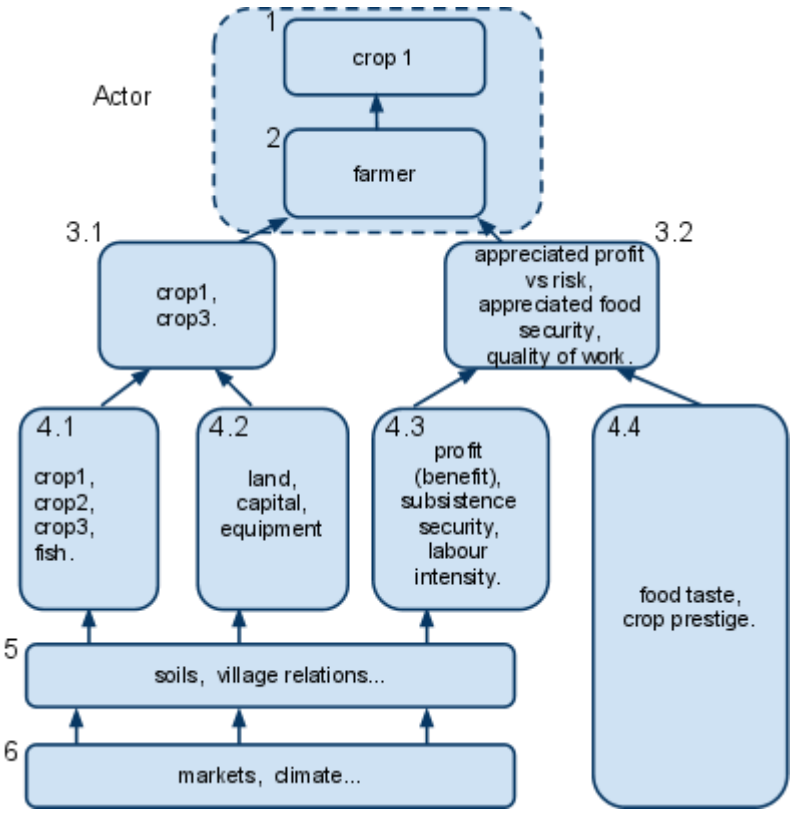

Fig. 3 The farmer description

fisherman but will formally describe his details later. The person "farmer" has more interest due to how tasty the food is and the prestige derived out of growing a particular crop (box 4.4) combined with farming results which include profit, subsistence security incase of low prices on the market, and labor intensity-the amount of human energy required to grow a crop (box 4.3). He also has a higher capital for farming i.e more land (acreage), equipment, credit, social capital etc the reverse being true for the fisherman as well. This is true in reality since farmers will go fishing if needed say during drought and fishermen would go farming if the fish stocks decline or there's a boom in selling crops.

In figure 3, we observe that before the farmer can execute an action from (box 4.1), he needs enough capital which is represented in (box 4.2) for it to be listed in (box 3.1). A list of all actions that can be done is found in (box 4.1) which include crops 1, 2, 3 and fish. The farmer then evaluates the attributes in (box 4.3) in relation to the appreciation attributed to food taste and prestige in (box 4.4) for all the options listed in (box 4.1). The result is represented in (box 3.2). Using the result from (box 3.2) combined with the options in (box 3.1), the farmer applies the involvement function and the maximum benefit heuristic, eqn (3), to determine the best option which in this case is crop1, as seen in box 1. Note here that we choose crop 1 but in reality, a combination (intercropping) of the best two crops (e.g crop1, 3) could be chosen for instance to avert drought risk.

Figure 3 is richer and provides more than just the basics and gives more elaborate explanation of the behavior of the agent. It contains the essentials and provides a working environment for the agent, therefore it is easier to reason about the agents by the domain experts.

Note also that from the figure, there are other factors which may influence the choice of actions due to the interaction with the environment as shown in component (box 5), where the farmer has control over factors like soils, village relations etc. while some factors are external to the actor like the markets, climate or international lending institutions like the World Bank, component (box 6), however the formal details of this is not discussed in this paper, as it is currently being worked on. A complete formal description is given below:

Agent Person

Specializes

Attributes
capital, knowledge: Conditional

Requires
True

Actions
sense
sc = True
bt = void

For the fisherman, we consider three parameters for the fishing activity; the time of fishing, the capacity of the boat, and the size of the fish net. Similarly for farming, it has three parameters that we use: the month of the year the crop is harvested, the area required for cultivation -the acreage and the output described in terms of the yield to determine the start conditions for both activities.

Agent Farmer

Specializes Person

Attributes
acreage (acre), labor intensity (calories/acre),
cost1, cost2, cost3 (dollar/acre),
yield1, yield2, yield3 (dollar/acre)
bt1, bt2, bt3 (dollars), subsistence security,
nutr1, nutr2, nutr3 (Joule/acre): Numeric
land, equipment, capital: Conditional

Requires capital ∧ land
Actions
crop1
sc = Env.Month ≤ March ∧ acreage ≥ 3 ∧ acreage × yield1 > 0.60 bt1 = acreage × (yield1 − cost1).

crop2
sc = Env.Month ≤ February ∧ acreage ≥ 5 ∧ acreage × cost2 < 0.70 bt2 = acreage × (yield2 − cost2).
crop3
sc = Env.Month ≥ May ∧ acreage ≤ 1
bt3 = acreage × nutr3.

Agent Fisherman
Specializes Person
Attributes
net size (inches), time(hours), catch(dollar/tonne),
capacity1, capacity2, capacity3 (tonnes),
bt1, bt2, bt3 (dollars)
cost1, cost2, cost3 (dollars): Numeric

equipment, capital, boat: Conditional

Requires capital ∧ boat

Actions

fish1
sc = netsize ≤ 0.75 ∧ capacity1 ≤ 5 ∧ time > 17 : 00

bt1 = (catch × capacity1) − cost1.

fish2
sc = netsize ≤ 0.5 ∧ capacity2 ≤ 10 ∧ 07 : 00 < time < 12 : 00

bt2 = (catch × capacity2) − cost2.

fish3
sc = 0.5 < netsize ≤ 1.5 ∧ capacity3 ≤ 15

bt3 = (catch × capacity3) − cost3.

6. CONCLUSION AND FUTURE WORK

In this paper, we have described the AiC$^+$ model an extension of the AiC framework by providing formal semantics which guide in explaining the complex behavior during optimal action selection by an agent in an environmental arena. From the example, we observe that using the agent type hierarchy, the agents "farmer" and "fisherman" specialize "person". We do not however consider the attributes of the fisherman here in the description framework. Using the involvement function Inv and the maximum benefit criteria leads us to the eventual action that is executed by the actor (farmer).

The main advantage of our model is that it is presented in a formal and non-ambiguous terms. According to Luck and d'Inverno [1995], formalization provides clarity in characterizing the nature of concepts. There is a demand of formal modeling with the need for implementation by providing clear and unambiguous definitions of state and operations on state which provide the basis for program development. We have explained how the AiC$^+$ model has a complex population schema in terms of an agent typing scheme. There's an agent hierarchy where

there are types and instances, the type inherits all the attributes of the instances sometimes referred to as subtypes. The relationship between the instances and the type is in the weighting scheme of the subtypes. This typing is dynamic in the sense that the instances can take on any action given the motivation and preference. The AiC^{+} also has a more fuzzy typing system where there is a fuzzy relation describing the type relation between an instance and the agent types defined in some agent type hierarchy. In this way we can, with ease, map a most specialized agent type with its "*pater familias*". We are yet to work on the formalization of different interaction schemes in the model which currently is under development. We are considering modes of interaction between the agents themselves and also between the agents and the environment.

REFERENCES

CHAINBI, W., JMAIEL, M., AND HAMADOU, A.B. 1998. Conception, behavioral semantics and formal specification of Multi-Agent Systems: Theories, Languages, and Applications. In the *4th Australian Workshop on Distributed Artificial Intelligence, pp. 16-28. LNCS vol. 1544*, Brisbane, Queensland, Australia.

DASTANI, M. 1998. 2APL: A Practical Agent Programming Language.JAAMAS.16 (3), 214-248.

DE GROOT, W.T. 1992. Environmental Science Theory. Concepts and methods in a one-world, problem-oriented paradigm at {https://openaccess.leidenuniv.nl/dspace/bitstream/1887/11548/1/11_511_207.pdf}.

DELOACH, S.A., AND VALENZUELA, J.L. 2007. An agent environment interaction model. In PADGHAM, L., ZAMBONELLI, F, Eds. AOSE 2006. Springer, Berlin. LNCS, vol. 4405, pp. 1-18.

ELSTER, J. 1989. Nuts and Bolts for the Social Sciences. Cambridge University Press, Cambridge.

FERBER, J., AND GUTKNECHT, O. 1998. A meta-model for the analysis and design of organizations in multi-agents systems. In DEMAZEAU, Y., (Ed.), ICMAS'98. pp. 128-135, Paris.

HUIGEN, M.G.A. 2004. First principles of the MameLuke multiactor modeling framework for land use change, illustrated with a Philippine case study. J Environ Manage. 72:5-21.

KENDALL, E. 1999. Role Modeling for Agent Analysis, Design and Implementation. In *1st International ASA/MA Symposium on Agent Systems and Applications.*

LUCK, M., AND D'INVERNO, M. 1995. A Formal Framework for Agency and Autonomy. In Proceedings of the *First International Conference on Multi-Agent Systems, AAAI Press, MIT pp. 254-260.*

OVERMARS, K.P., DE GROOT, W.T., AND HUIGEN, M.G.A. 2007. Comparing Inductive and Deductive Modeling of Land use Decisions: Principles, a Model and an Illustration from the Philippines. Hum Ecol 35, 439—452.

RAO, A. S., AND GEORGEFF, M. P. 1995. BDI Agents: From Theory to Practice, Techinical Note, ICMAS.

ROUCHIER, J., BARRETEAU, O., AND BOUSQUET, F., PROTON, H. 1998. Evolution and co-evolution of individuals and groups in environment. IN DEMAZEAU, Y., (Ed.), ICMAS'98. pp. 254-269, Paris.

VAYDA, A. P. 1983. Progressive Contextualization: Methods for Research in Human Ecology. Human Ecology 11 (3), 265-281.

VAYDA, A. P., AND WALTERS, B. B. 1999. Against Political Ecology. Human Ecology 27 (1), 167-179.